Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2004 Proceedings

Americas Conference on Information Systems (AMCIS)

December 2004

Inside Peer-to-Peer - Some Theory and an Empirical Analysis of a File Sharing Network

Nick Gehrke
University of Goettingen

Lutz Seidenfaden
University of Goettingen

Follow this and additional works at: http://aisel.aisnet.org/amcis2004

Recommended Citation
Gehrke, Nick and Seidenfaden, Lutz, "Inside Peer-to-Peer - Some Theory and an Empirical Analysis of a File Sharing Network" (2004). *AMCIS 2004 Proceedings*. 502.
http://aisel.aisnet.org/amcis2004/502

This material is brought to you by the Americas Conference on Information Systems (AMCIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in AMCIS 2004 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

Inside Peer-to-Peer – Some Theory and an Empirical Analysis of a File Sharing Network

Nick Gehrke

Institute of Business Informatics, Dep. II
University of Goettingen, Germany
ngehrke@uni-goettingen.de

Lutz Seidenfaden

Institute of Business Informatics, Dep. II
University of Goettingen, Germany
lseiden@uni-goettingen.de

ABSTRACT

Peer-to-Peer (P2P) File sharing Applications are responsible for a large amount of the (illegal) distribution of media content which systematically causes copyright infringements. In order to estimate a possible economical influence, it seems to be feasible to analyse the content of those systems. Firstly, the paper on hand describes theoretical P2P architectures in order to shed light on the possible configuration of file sharing networks. Secondly, we describe which technical means are necessary for an automated analysis of popular applications like Gnutella or Kazaa and how this analysis is conducted. Subsequently, we present first simple statistics to demonstrate what reports can be run on data collected from the aforementioned systems.

KEYWORDS

Peer-to-Peer, Peer-to-Peer Architectures, File sharing, Statistical Analysis, giFT

INTRODUCTION

Peer-to-Peer (P2P) systems are highly decentralized systems in which the nodes (or peers) increase the overall potential of the system by contributing content or services. Although P2P comprises a variety of possible applications (e.g. Instant Messaging and Grid Computing) the paper on hand focuses on file sharing applications due to their popularity. The rapid growth and popularity of file sharing networks is mainly caused by the high value of these networks to its users. This can be seen as a result of Metcalfe's law, which states that the network value rises by the square of the number of users (Leuf 2002). However, the aim is not to examine the economical influence of P2P-file sharing on the music industry (see Oberholzer/Strumpf 2004), but rather to show the theoretical architectures of those systems and which information can be obtained automatically from them. Furthermore, we show which basic clues can be derived from that data. Kazaa is probably the best known application in the field and therefore our first choice for the analysis. Initially, we explain theoretical foundations of P2P networks. Subsequently an approach for the automatic data retrieval from the system without using the Kazaa software is described and demonstrated. For that purpose an application ("Kazaa Spy") has been developed. After the collection of a certain amount of data, some simple statistical analyses are performed which e.g. give information about the quantity of users and content. During the analysis the focus is put on the distribution of music files since that type of media is widely available within the Kazaa system. Looking from the limited perspective of a peer, we identified the following questions to be answered:

- Is the search within Kazaa performed efficiently or is the same local search result returned often?
- How many different versions of the same music title co-exist and how are they distributed throughout the system?
- How many versions of the same music title does a user provide?

P2P IN THEORY – NETWORK ARCHITECTURES

At this stage theoretical P2P network architectures are described and categorized. A definition of the term P2P is not provided here but can be found in (Miller 2001, Barkai 2001, Shirky 2000). P2P networks are not structured the same way, in fact a lot of degrees of freedom exist while constructing such a network. Therefore a classification seems to be necessary (Minar 2001a, Minar 2001b). P2P architectures can be categorised in the types atomistic (or pure), user-centric and data-centric (Leuf 2002). An atomistic structure consists only of peers and no central server exists. In contradiction, the two latter

models comprise at least one central instance for coordination purposes. In a user-centric network, the central server only contains an address list of all available peers within the network while in a data-centric system additional information regarding content is stored on the central server. Unfortunately the categories mentioned above do not describe existing systems sufficiently. From an architectural point of view there is no real difference between user- and data-centric systems because it does not matter what kind of tasks the central server does perform. The important point is that a central instance exists. Therefore, a different classification of existing systems should be followed, which divides systems into (Hong 2001):

- pure P2P architectures,
- brokered P2P architectures, which have a central unit and
- hybrid and hierarchical architectures.

Pure P2P

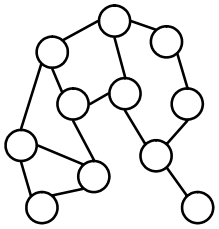


Figure 1: Pure P2P architecture

In a pure P2P environment the highest possible degree of decentralisation can be found. There is no such thing as a central unit for coordination purposes. As a result, search queries need to be passed on from one peer to another (forwarding). Caused by the forwarding, which increases exponentially if a peer forwards the same query to more than only one other peer, pure P2P systems have high demands on network capacity. In the past trouble caused by this issue could be observed within the Gnutella system (Ripenau et. al. 2002). In the meantime search algorithms with better than exponential scalability have been developed (Aberer 2001). Another issue is the possibility that a search only displays results of a small fraction of the network, because the quick distribution of a query leads to a shallow search depth. The big advantage of a pure architecture is robustness against authorities, which cannot easily shut down such a network due to the lack of a central unit. This might be attractive for illegal file sharing, but in a commercial P2P environment this advantage is questionable.

Brokered P2P

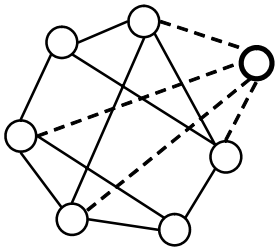


Figure 2: Brokered P2P architecture

To circumnavigate the issues regarding performance and scalability, the brokered architecture is coordinated by a central server. This ensures a faster discovery of peers and content. However, the server does not provide resources such as content or disk space; it only provides coordination mechanisms. The central unit comes in two flavours (Dreamtech 2002):

As a *Discovery-Server* (user-centric) whose only task is to help find peers. Therefore it maintains a list of available peers.

A *Lookup-Server* extends the search capability of a Discovery-Server with the ability to find resources and services. A good example for such a server is the index server of the file sharing application Napster on which information about peers and their content was stored.

Brokered P2P networks seem to be more relevant for commercial P2P applications than a pure P2P concept, because a relatively quick search for resources and services can be ensured through a central unit. In addition, it saves network bandwidth because the central unit is contacted directly rather than forwarding queries from peer to peer. Data that should only be manipulated by authorized peers (master data, account balances or public keys) can be maintained more easily on a central unit than on distributed peers. Furthermore, the central unit can also be used as a billing service or a Trust Centre for applications relying on asymmetric encryption.

Hybrid and hierarchical P2P

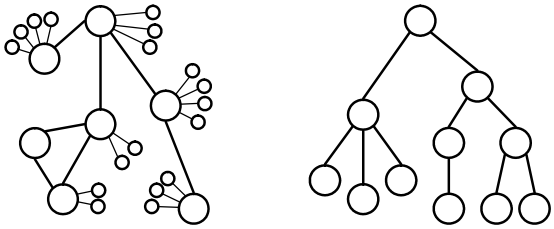


Figure 3 : Hybrid (left) and hierarchical (right) P2P architecture

Architectures with and without a central unit do not mark alternative concepts. It is possible and often reasonable to combine both in order to bring the advantages of complete decentralisation and a central unit together. In contrast to a central approach a hybrid architecture comprises a couple of equal peers which share coordination tasks ("supernodes" (Hong 2001)). The allocation of supernodes can be spontaneous e.g. a peer with good performance and high availability can become a superpeer temporarily. As a result clusters with local coordination units emerge. Search queries are no longer passed from peer to peer but are handled by the local superpeer which contacts the superpeers in its neighbourhood if necessary. This architecture avoids a central and therefore vulnerable unit and is successfully used within Kazaa.

In hierarchical architectures the participating peers are not equal. They are all arranged in a predefined hierarchy in which some nodes can be superpeers, serving a couple of normal peers. Likewise within hybrid architectures, the search functionality is the responsibility of the superpeers. Problems can arise if a superpeer along the way from top down is damaged. This will cause several branches of the network to be no longer available. A possible solution is a redundant organisation of hierarchies.

Independent of its architecture, a P2P Network can be organized in a structured or unstructured manner. Unstructured networks, while not centrally planned in structure, grow according to a simple self-organizing process. Prominent examples are the file sharing networks Gnutella, Freenet (Adamic et.al. 2002) and KaZaa. In contrast, in structured networks a certain logical structure ("overlay") is maintained regardless of the size and the type of the (underlying) network. An example would be a P2P-network organized by the chord protocol (Stoica et al. 2001, Dabek et al. 2001) which always maintains a ring like structure, the so called chord ring. In a structured file sharing network a certain song - if available on one of the nodes - will always be found (quite fast, i.e. O(log n) for the Chord algorithm), while in an unstructured system there is no guarantee to find it in limited time.

TAPPING KAZAA AND GNUTELLA

After having described the theoretical background we now turn to practice. In this paragraph a method for the automated search and analysis of file sharing networks like Kazaa and Gnutella is described. While tapping the open Gnutella protocol is quite easy, Kazaa uses the proprietary and secret Fasttrack protocol. Despite this fact it is possible to develop a simple client application for both networks.

The giFT Project

giFT (giFT:Internet File Transfer) is an Open Source project (giFT 2004). The aim of the project is to build one interface for different P2P networks. So far, it incorporates Fasttrack, Gnutella and OpenFT (gift project protocol). The architecture of the application is divided into two components: the giFT daemon and a giFT client. The daemon is a standalone application which does not need to be modified by the developer.

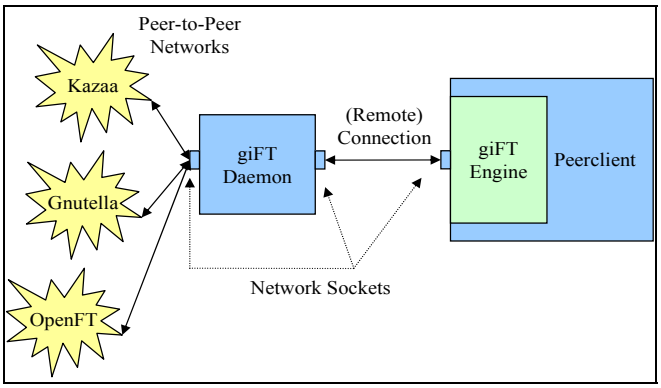


Figure 4: The giFT architecture

It lacks a GUI and is purely responsible for the communication with the underlying P2P network(s). It can be understood as a proxy that can also be contacted by remote clients. The client has to be programmed by a developer. The communication capability between giFT and the client is encapsulated in C++ class files, the so called giFT engine. As a result, the developer doesn't need to know anything about the underlying protocol. Figure 4 depicts the giFT architecture.

Scanning Kazaa

Using giFT it is possible to scan the Kazaa network from a peers' view. In this paper Kazaa has been chosen as an example due to its popularity and vast number of users. Utilizing the giFT engine, it is easy to perform a search for certain files within Kazaa. One needs to feed the keywords into the engine and collect the results after the search is finished. The results can be used in different ways (e.g. display on screen, write to logfiles or databases). Figure 4 shows the complete source code necessary to perform a search for content (music titles in our case) in the Kazaa network.

```
#include <vcl.h>
#include <iostream.h>
#include <conio.h>
#include <time.h>
#include <stdio.h>
#include <dos.h>
#include "Engine.h"                              // include giFT Engine

#pragma hdrstop
#pragma argsused

using namespace KCeasyEngine;
typedef list<TSearchResult*>::iterator TResultIterator;
int main(int argc, char* argv[])
{
        string keyword="Madonna";
        string networkstring="FastTrack";
        int filetype=SRAudio;
        TEngine* Engine=NULL;
        TSearch* Search=NULL;
        int number=0,numberlast=0;

        Engine = new TEngine("KazaaSpy","0.8");  // initialize Engine
        Engine->Init();
        cout << "Engine started...\n";

        Engine->TurnOnline("127.0.0.1",1213);
        cout << "Attempting to go online...\n";

        cout << "Press any key!\n";
        while(getch()==0){}

        if(Engine->IsOnline()) cout << "Engine is online...\n";
```

```
            else cout << "Sorry, Engine is offline!\n";
         if(Engine->IsOnline()){
            Search = Engine->NewSearch(keyword,dateitype,networkstring);
            cout << "Query...Keyword: "<<keyword<<"\n";
            Search->Start();
            cout << "Query started\n";
         }
          // Wait until search is finished
          while(Search->GetState()!=2){
             number=Search->NoOfResults();
             if (number!=numberlast)
                cout << "Searching: "<<Search->NoOfResults()<<" files found.\n";
             numberlast=number;
         }
         TResultIterator Result=Search->GetResultsBegin();

         // iterate through search results
         for(int i=0;i<Search->NoOfResults();i++){
              TSearchResult* r=Result.operator *();
              cout << "Found: "<< r->FileName<< ", "<< r->UserId <<"\n";
              Result.operator ++();
         }
         cout << "Search finished. Press any key.\n";
         while(getch()==0){}
         return 0;
}
```

Figure 5: Sourcecode for a simple version of Kazaaspy (written and compiled with Borland C++ Builder 6.0)

The application mentioned above could return for example a result as shown in Figure 6. It only shows the filenames of the files the application found by searching for a specified keyword and users which are in possession of those files. It would be childs play to obtain much more information but that is not in scope at this stage of our examination.

```
Engine started...
Attemtping to go online...
Yeath, Engine is online...
Query … Keyword: Madonna
Query startet
Searchstate: 1
Lap Number : 0
Searching: 23 files found.
Searching: 42 files found.
...
Searching: 209 files found.
Searchtime (Seconds): 5
Listing results:
Found: Various Artists - Rufio - Like A Prayer.mp3, hate_breed@216.239.94.32
Found: 02-hollywood-wax.mp3, hate_breed@216.239.94.32
Found: Madonna - Take A Bow.mp3, defaultuser@192.168.1.101
Found: PAZ MARTINEZ Verßs de Maddona.mp3, msilvetti@200.45.30.42
Found: Madonna - Die Another Day .mp3, kim@209.53.195.240
Found: Madonna - Don't Tell Me (Club Remix).mp3, MonteC97@192.168.0.2
Found: Top 40--Madonna- Music.mp3, nicoletruscott@209.89.145.36
Found: Madonna - Music (remix).mp3, MonteC97@192.168.0.2
Found: Madonna - Music.mp3, blondie2415@205.251.206.145
Found: Madonna - Vogue (1).mp3, KazaaLiteK++@192.168.2.39
Found: Modonna - Vogue.mp3, KazaaLiteK++@192.168.2.39
Found: Madona - Material Girl.mp3, kim@209.53.195.240
Found: Dexter Freebish--06-My Madonna.mp3, SuperFly351@192.168.1.100
Found: Madonna - Santa Baby.mp3, princesslaura@192.168.1.103
Found: Madonnd-beautiful stranger (instrumental).mp3, raivaldo@200.221.55.97
Found: Madonna - Papa Don't Preach.mp3, blondie2415@205.251.206.145
Found: Madonna_-_Die_Another_Day.mp3, blondie2415@205.251.206.145
```

```
Found: Madonna - Like a Prayer.mp3, inthemiddle@192.168.1.100
Found: madonna - ray of light (william orbit liquid mix).mp3, princesslaura@192.168.1.103
Found: Madonna - Material Girl.mp3, jenniferreece@172.16.1.33
Found: Madona American Live.mp3, Todesengel@82.83.199.5
...
Press Key to exit...
```

Figure 6: Results of a search request with the KazaaSpy application

WHAT DOES KAZAA REVEAL?

After the explanation how access to the Kazaa network works, we now want to shed light on gathering information about detectable users and content within Kazaa. On one hand this is interesting for Kazaa users wondering how anonymous they are (i.e. what personal information is divulged within KaZaa) while uploading and downloading files. On the other hand music labels could be interested which user data can be derived from KaZaa in order to bring them to court or to use that data for marketing purposes. Table 1 depicts what a data set of a result reveals within a Kazaa search request.

Information	Title	Example
FileName	Title of the file.	05 - Shut Up (1).mp3
FileSize	Size of the file.	3564003
MimeType	Mimetype of the file.	audio/mpeg
FileType	Type of the file.	241
Availability	Availability of the file.	1
UserName	Username within Kazaa	MasterP@65.35.37.250
UserId	User-ID within the Kazaa network.	MasterP@65.35.37.250
SourceId	Unique information where to find the file (like an URL).	FastTrack://65.35.37.250:0/=F/KC3tGOSKw3G-bpsxIGzRuK32Dw=?shost=65.33.131.61&sport=4215&uname=MasterP
DataId	Unique identification of the file (e.g. hash value).	FTH:=F/KC3tGOSKw3GbpsxIGzRuK32Dw=
MetaData	Metadata of the file found (depends on file type).	

Table 1: Information revealed in a data set of a Kazaa result (Search words were: "Black Eyed Peas Shut Up").

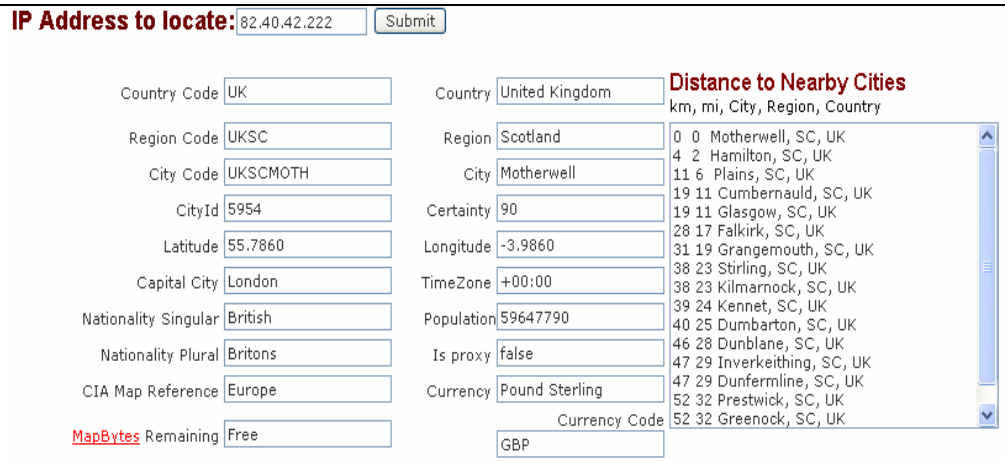
As one can see it is easy to obtain the IP address of a user. In order to find out about redundancies, hash values (DataId) of the files help to identify redundant files. Due to this hash value of a file it is possible to conduct a download from different peers at the same time (multipeer-download).

ANALYSIS OF USERS AND CONTENT

In the following automatically accomplished search requests to the Kazaa network are (statistically) evaluated. The results can help to estimate the dimension of illegal copy processes of media products, e.g. music or films, within a file sharing system. This information is interesting for copyright holders, especially for the big labels like Sony or BMG.

How anonymous is a Kazaa User?

In order to reveal a user's identity one has to examine the IP of a Kazaa user. The IP address is provided by the underlying protocol interpreted by the giFT engine. Thus it is easy to identify the user's internet service provider (ISP) or the organisation he belongs to (if the user uses a static IP address).

Requesting the service www.geobytes.com	

nslookup request	`nslookup 82.40.42.222`
	`Name: 82-40-42-222.cable.ubr06.uddi.blueyonder.co.uk`
	`Adress: 82.40.42.222`

Figure 7: Finding out the location and the provider/organisation of a Kazaa user.

The owner of the IP address can be found out easily by using the (windows) command nslookup. Furthermore there are some services offered through the internet, which enable one to locate an IP geographically on a world map (providing the degree of longitude and latitude, see for example www.geobytes.com/IpLocator.htm?GetLocation). Figure 7 shows the results of such requests.

It is obvious that it is very easy to retrieve some basic information about a Kazaa user with internet services free of charge. Finally the personal identity of a Kazaa user is not detectable with these kinds of methods. For this purpose the ISP of the Kazaa user has to reveal which person (respectively which modem connection) was provided with the IP in question. Normally, this information underlies data protection laws. But the decision which groups of persons are allowed to know about this information is subject to the respective national legislations.

STATISTICAL ANALYSIS

General Results

In order to conduct statistical analyses it is important to focus on a special content type. In the following we focus on music files due to the up-to-dateness of this topic. We examined music titles taken from the MTV Euro Top 20 Charts of February 1st, 2004. In order to conduct an appropriate data ascertainment we sent search requests into the Kazaa network during a time period of about 17 hours. We sent keywords one after the other for all 20 music titles 115 times, so for every music title we obtained 115 result sets. We logged every data set in a file. After that all data sets have been imported in a database and we started the examination. In order to give a compact view on our findings, we only show results of 10 music titles. The data collection was conducted from a computer in the intranet of our university. For our examination we formulated the following questions:

1. Do several search requests with the same keywords often return results with similar users?
2. Do several search requests with the same keywords often return results with similar files?

The two questions address the possibility that search results only lead to users and content of the local surrounding of the requesting peer. The opposite possibility would be that search results reflect the richness of the whole network. Table (2) shows the results for 10 randomly chosen music titles.

Keywords	Number of results	Average number of results per round	Different users found	Different users found/Number of results	Different files found	Different files found/Number of results

Alicia Keys You Know My Name	19546	169,97	9976	51,04%	1265	6,47%
Beyonce Me Myself	22324	194,12	11052	49,51%	2116	9,48%
Black Eyed Peas Shut Up	11213	97,50	3230	28,81%	1108	9,88%
Britney Spears Madonna Me Against Music	11207	97,45	5818	51,91%	675	6,02%
Christina Aguilera Voice Within	23036	200,31	10173	44,16%	1994	8,66%
Dido White Flag	9904	86,12	3009	30,38%	1351	13,64%
Evanescence My Immortal	19122	166,28	7909	41,36%	2919	15,27%
No Doubt My Life	20851	181,31	9849	47,24%	1727	8,28%
Pink Trouble	12247	106,50	3682	30,06%	1895	15,47%
Sarah Connor Music Is The Key	3448	29,98	1889	54,79%	309	8,96%

Table 2: Important result of 10 music titles of the MTV Euro 20 charts.

Table 2 shows that all in all a considerable number of search results has been collected during the survey period (see column 2). The third column reports the average number of returned data sets during one request for each music title.

In order to answer question 1 column 4 shows the number of different users found for each music file for all 115 search requests (thereby, a user is a combination consisting of an IP and an arbitrary user name). Column 5 calculates the ratio of the number of different users to the number of search results. For the first music title the ratio is about 50 percent. Thus, every user has been found 2 times on average. If the search requests would have produced always similar results, the amount of different users would be much lower and should produce the dimension of column 3. Obviously, not only the local surrounding of the requesting peer is scanned. Thus, the results are reflecting the heterogeneity of the whole network or at least of a wide range of the surrounding. Theoretically, the high number of different users could be the consequence of high frequent login and logout processes of the peers. However, this effect cannot be verified, but must be considered to be improbable.

Column 6 can be used to answer the second question. This column shows the number of different files found for each music title. As one can see the number of different files found is relatively high. Thus, very many different versions of a music title are coexisting. A version of a file can be distinguished from another version by a different hash value. Hereby one has to notice that smallest changes of the file (e.g. changes of the ID metatag information) result in a different file hash value. It is possible that the high number of different versions is caused by the infiltration of so called "junkfiles" by the music industry in order to pollute the music enjoyment of music downloaders. Answering the two aforementioned questions, we state that the Kazaa network is thoroughly scanned and thus provides well diversified results in terms of both users and content.

Concentration of music file versions

In the preceding section was shown that a high number of different versions of a music title exists. The next question arising is: Are these different versions distributed uniformly within the Kazaa network or are some versions widely distributed and thus dominate the search results? In order to answer this question one can consult the theory of network effects (Shapiro et.al. 1999). This theory states that one music title which is more widely spread than another is further spread with a higher growth rate and thus leading to a polarisation of versions. Because one file is more widespread than another, search requests will often return the more widespread file so that further dispersion accelerates. In order to verify if this particular theory sufficiently predicts empiricism, we constructed concentration curves of the versions of a music title. Figure 8 shows the concentration curves of versions of the two titles "Christina Aguilera – The Voice Within" and "No Doubt – It's my Live".

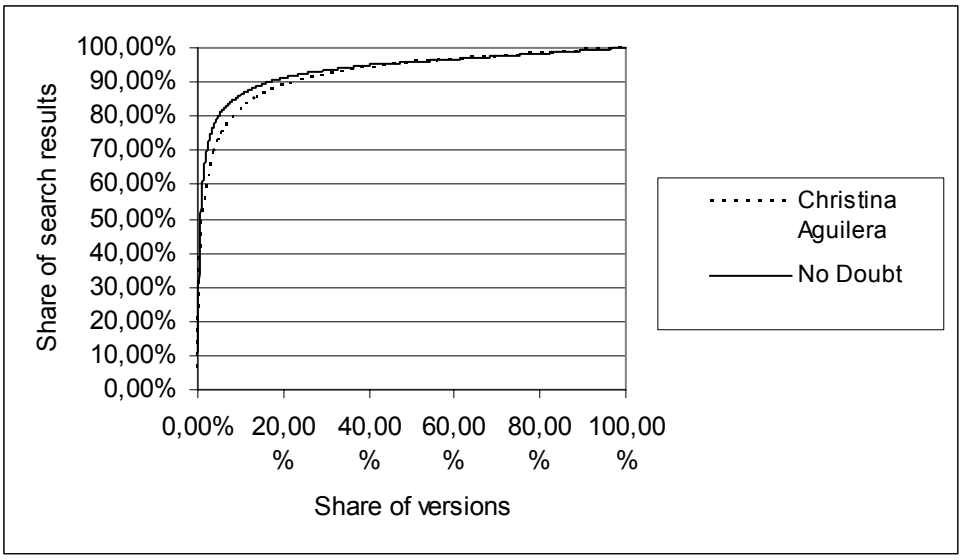


Figure 8: Concentration curves of two music titles.

It is immediately obvious that a very small share of versions covers a very high amount of the search results. The curve progression is similar for other music titles and can be considered to be a general phenomenon. If the versions were distributed uniformly within the network, the curve progression would look like a bisecting line. But in the case on hand only 3-4 percent of the existing versions cover 70 percent of the search results. Due to this result one can assess the strategy of the music labels, which try to put so called "junkfiles" into the network. Due to the strong polarisation of versions a fast selection of bad versions will take place leading to a fast erasure of "junkfiles" by the users. At the end of the process "junkfiles" will not distribute very well so that it seems likely that such a "junkfile" strategy is not very helpful at all.

Individual Supply of Music Titles

After the presentation of the polarisation phenomenon, we proceed by evaluating the users' supply of different versions of one music title within the network. Thereby one can expect that usually a user would only provide one version of a music title, because he is only interested in the consumption of music and further versions of a music title do not provide an additional value to the user. Figure 9 shows, what percentage of the users possess how many versions of a music title. It also shows the amount of versions distributed among the users for one exemplary title. The curve progression is similar for other music titles and can thus be considered to be a general phenomenon. As one can see 85 percent of the users only possess one version of a music title. After that, the share of users possessing more than one version strongly declines. In the example only one user possess 12, 18, 21 and 22 different versions of the title.

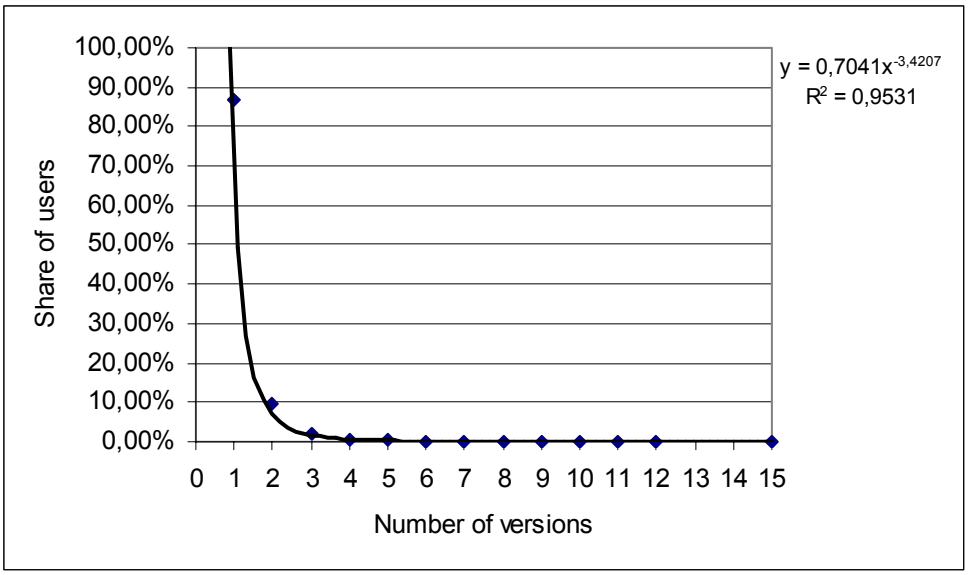


Figure 9: How much percent of the users posses how many different versions of a music title (Title here is: "Alicia Keys – You Don't Know My Name")?

The phenomenon that a very small number of users provide a huge number of versions is very typical for other music titles as well and can be regarded to be a general phenomenon. As a result, the examination shows that normally a user has only one version of a music title, but there are some very rare hubs providing a huge number of versions to the network. The statistical distribution of the amount of versions has been fitted with a power regression in figure (9). The coefficient of determination is about 95 percent so that the regression fits the empirical data very well. The distribution of the amount of versions can be described by a so called power law (Barabasi 2002).

CONCLUSION

The paper on hand analysed the file sharing system Kazaa. As a result the following answers to the questions formulated in the introduction can be given:

- The Kazaa network is well scanned when identical search requests are send in the network one after another. The search results show heterogeneity concerning users and content.
- There is a huge amount of different versions of a music title. The dispersion of the versions is strongly polarised. A small number of versions are widely spread; most versions do not have a broad dispersion. Strategies that aim to disturb file sharing, like "junkfile injection" seem to be an unsuccessful approach.
- Users usually possess only one version of a music title. Thereby, the amount of versions possessed by a user follows a power law. There is a very small number of users possessing a high number of different versions.

Future research activities should focus on the estimation of diffusion curves (Schoder 1995) of music titles. This would address the diffusion of a music title over time. But for this purpose one has to identify appropriate content very early in order to observe the whole dispersion cycle of the music title in question. Another interesting aspect not focussed in this paper is the "freeloader"-problem. Freeloaders are users who only download content but do not provide (or only a minimum) content themselves. An examination of that problem would clarify the structure of the file sharing network.

REFERENCES

1. Aberer, K. (2001): Aberer, K.: P-Grid: A self-organizing access structure for P2P information systems, in: Sixth International Conference on Cooperative Information Systems, Lecture Notes in Computer Science 2172.
2. Adamic et al. (2002): Adamic, L., Lukose, R., Huberman, B.: Local Search in Unstructured Networks, URL: http://www.hpl.hp.com/research/idl/papers/review/reviewchap.pdf, 2002
3. Barabasi, A. (2002): Barabasi, A.: Linked – The New Science of Networks, Cambridge.
4. Barkai, D. (2001): Barkai, D.: P2P Computing - Technologies for Sharing and Collaborating on the Net, Hilsboro.

5. Dabek et al. (2001): Dabek F., Brunskill, E., Kaashoek, F., Karger, D., Morris, R., Stoica, I., Balakrishnan, H.: Building Peer-to-Peer Systems With Chord, a Distributed Lookup Service, *Proceedings of the 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII),* 2001
6. Dreamtech (2002): Dreamtech Software India: P2P application development: cracking the code, New York.
7. giFT (2004) giFT: Internet File Transfer, http://gift.sourceforge.net, date 21.02.04
8. Gnutella (2000): The Gnutella Protocoll Specification v0.4, www9.limewire.com/developer/gnutella_protocol_0.4.pdf, date 24.03.2003.
9. Hong, T. (2001): Hong, T. : Performance, in: Oran, A.: P2P - Harnessing the Power of Disruptive Technologies, Sebastopol 2001, S. 203-241.
10. KaZaa (2003): KaZaa Homepage, www.kazaa.com, date 23.04.2003.
11. Leuf, B. (2002): Leuf, B.: Peer to Peer: Collaboration and Sharing over the Internet, Boston.
12. Minar, N. (2001a): Minar, N.: Distributed Systems Topologies: Part 1, www.openp2p.com/lpt/a//p2p/2001/12/14/topologies_one.html, date 14.12.01.
13. Minar, N. (2001b): Minar, N.: Distributed Systems Topologies: Part 2, http://www.openp2p.com/pub/a/p2p/2002/01/08/p2p_topologies_pt2.html, date 23.04.2003.
14. Oberholzer/Strumpf (2004): Oberholzer, F./Strumpf, K.: The Effect of File Sharing on Record Sales – An Empirical Analysis, URL: http://www.unc.edu/~cigar/papers/File sharing_March2004.pdf, date 31.3.2004
15. Ripeanu (2002): Ripeanu, M./Foster, I./Iamnitchi, A.:Mapping the Gnutella network: Properties of largescale P2P systems and implicatons for system design, *IEEE Internet Computing Journal,*6,1, 50-57.
16. Schoder (1995): Schoder, D.: Erfolg und Misserfolg telematischer Innovationen – Erklärung der „Kritischen Masse“ und weiterer Diffusionsphänomene, Wiesbaden.
17. Shapiro/Varian (1999): Shapiro,C./Varian, H. R.: Information Rules, Harvard
18. Shirky (2002): Shirky, C.: what is p2p ... and what isn't, www.openp2p.com/pub/a/p2p/2000/11/24/shirky1-whatisp2p.html, date 07.08.2002.
19. Stoica et al. (2001): Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, R.: Chord: A scalable peer-to-peer lookup service for internet applications, *Applications, Technologies, Architectures and Protocols for Computer Communication Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications San Diego*, *2001*, 149 – 160